Prolog toets 2: 20 April 1998

Beoordeling: aantal punten /100

vraag	aantal punten
1a	10
1b	15
1c	15
2a	10
2b	10
2c	10
3	16
4a	6
4b	4
4c	4

Vraag 1: Problem solving strategie

Het 8-blokjes puzzle probleem is:

De puzzle heeft een speelveld van 3 bij 3, met 8-blokjes (schuifjes) genummerd 1 t/m 8, en 1 leeg veld. Een schuifje kan alleen op de lege plek geschoven worden als het aan het lege veld grenst. Het lege veld kan dus als het ware rondbewegen, door het verwisselen van de lege plek met een aangrensde schuifje. De schuifjes (blokjes) mogen alleen horizontaal en verticaal bewegen op het 3 bij 3 speelveld. De gewenste eindtoestand van het spel is dat alle schuifjes in een oplopende volgorde staan, als je rij1, rij2 en rij3 achtereen volgend leest.

Voorbeelden van goede eindsituaties zijn:

1	2	3
	4	5
6	7	8

1	2	3
4	5	6
7	8	

Het 8-blokjes puzzle probleem is: gegeven een willekeurig begintoestand van het speelveld, hoe kan je dan van de begintoestand tot een eindtoestand komen? Dit probleem kan beschouwd worden als een zoekprobleem, waarbij we een basis -zoekstrategie kunnen toepassen (bijv. depth-first).

Een zoekprobleem moet dan gekarakteriseerd worden door:

- toestandsruimte
- begintoestand
- goal conditie wat een eindtoestand weer geeft

Vraag 1a: Geef een representatie in Prolog voor een toestand (situatie) in de zoekruimte voor het 8-blokjes puzzle probleem.

Hoe wordt in je gekozen representatie het volgende bord gerepresenteerd?

7	8	3
	4	5
1	2	6

Vraag 1b: Schrijf een "goal" conditie voor het 8-puzzle probleem. Deze conditie moet dus uitdrukken wanneer een bord een gewenst eindbord is. Gebruik hierbij je gekozen bord-representatie. Schrijf ook commentaar bij de code voor "goal".

Vraag 1c: Geef de definitie van de successor relatie, zowel in woorden als in Prolog.

Vraag 2: Frames / Semantische Netwerken

Gegeven zijn de volgende frames:

```
FRAME: bird
a_kind_of: animal
moving_method: fly
active_at: daylight

FRAME: albatross
a_kind_of: bird

FRAME: kiwi
a_kind_of: bird
moving_method: walk
active_at: night
```

Deze frames zijn in de volgende Prolog clauses vertaald:

```
bird(a_kind_of,animal).
bird(moving_method,fly).
bird(active_at,daylight).
albatross(a_kind_of,bird).
albatross(moving_method,fly).
albatross(active_at,daylight).
kiwi(a_kind_of,bird).
kiwi(moving_method,walk).
kiwi(active_at,night).
```

Verder wordt de volgende procedure voor "value" gebruikt:

```
value( Frame, Slot, Value):-
   Query =.. [Frame, Slot,Value],
   call(Query), !.
value( Frame, Slot, Value):-
   parent( Frame, ParentFrame),
   value( ParentFrame, Slot, Value).
parent( Frame, ParentFrame):-
   Query =.. [Frame, a_kind_of, ParentFrame],
   call(Query).
```

Vraag 2a: Welke Prolog clauses uit de vertaling van de frames zijn niet strikt nodig om de juiste antwoorden voor "value(+Frame,+Slot,-Value)" queries te krijgen? (gegeven de code van de procedure value)

Vraag 2b: Wat is het effect van deze "overbodige clauses" op de "value(..)" queries?

Vraag 2c: Kun je de value procedure aanpassen zodat eerder berekende value oplossingen toegevoegd worden in de database (programma)?

Vraag 3: Prolog Inbuilt procedures

Wat is het antwoord van de Prolog interpreter?

```
1: ?- X =[1,2], var(X).
2: ?- nonvar(X), X=toets.
3: ?- punt(1,X)=..L.
4: ?- X=lijn(punt(1,3),Y), X=..[Z|Zs].
5: ?- punt(X,Y)==punt(1,2).
6: ?- punt(1,2)=punt(X,Y).
7: ?- X=2, (3*X) =\= (4+X).
8: ?- X=2, (3*X) \== (4+X).
```

Vraag 4: Assert / Retract

Vraag 4a: Wat is het antwoord van de Prolog interpreter? Ga er van uit dat de Prolog-database voor iedere vraag leeg is.

```
1: (start met lege Prolog-database)
   ?- asserta(p(1)),asserta(p(2)),asserta(p(3)).
      yes.
   ?- p(X).

2: (start met lege Prolog-database)
   ?- assert(p(1)),assert(p(2)),assert(p(3)).
      yes.
```

```
   ?- retract(p(X)).

3: (start met lege Prolog-database)
   ?- assert(p(1)), assert(p(1)), retract(p(1)).
      yes.
   ?- p(X).
```

Vraag 4b: Schrijf een Prolog procedure "laad-feiten". Laad-feiten heeft 1 argument, namelijk een lijst met feiten. Deze feiten moet de procedure laad-feiten in de Prolog-database laden.

Dus: laad-feiten([p(1),p(2),p(3)]) heeft als resultaat dat in de Prolog database de feiten p(1), p(2), en p(3) staan.

Vraag 4c: Noem een nadeel van het gebruik van assert en retract?